A Hybrid System for Network Traffic Forecast Using Firefly and Cortical Learning Algorithm

K. C. Eme[1] , V.I. Anireh[2], & N. D. Nwiabu[3]
[1,2,3]Department of Computer Science,
Rivers State University, Port Harcourt, Nigeria
DOI: 10.56201/ijcsmt.v10.no3.2024.pg105.120

Abstract
Network traffic forecast is a critical aspect of network management and cybersecurity, encompassing the prediction and analysis of data transmission patterns within computer networks. As the volume and complexity of network traffic continue to increase exponentially, accurate forecasting becomes indispensable for ensuring efficient resource allocation, optimizing network performance, and detecting potential security threats. Object Oriented Analysis and Design (OOAD) was adopted as the research methodology, and python was used as the programming language. An experiment was conducted to detect malicious traffic on network systems using a hybrid approach of firefly algorithm and cortical learning, encompassing two phases: Exploratory Data Analysis (EDA) and training of the Random Forest Classifier. In the EDA phase, techniques were employed to address dataset imbalance in the NSL-KDD dataset through random oversampling, alongside identifying the ten most important features using Isolation Forest. During the model training phase, parameters were initialized for both algorithms, leading to iterative optimization to achieve an optimal balance between exploration and exploitation crucial for capturing complex network traffic patterns. Evaluation of the model's outcomes, including training and testing results, was conducted using classification reports and confusion matrices. The model showcased promising results, achieving an accuracy of 99.9% in detecting malicious network traffic, and significantly outperformed existing systems, demonstrating the efficacy of the hybrid approach in network intrusion detection. This study underscores the effectiveness of the hybrid approach, offering superior performance compared to traditional methods.

Keywords- *Network Traffic Forest, Cortical Learning, Intrusion Detection, Machine Learning*

1. Introduction

The prediction of network traffic for the purpose of intrusion detection is an essential field of study within the realm of cybersecurity. Scholars are employing machine learning and deep learning methodologies to construct resilient systems with the ability to detect and alleviate several forms of attacks on communication networks. Intrusion detection systems (IDS) are of considerable importance in the surveillance of network traffic with the purpose of identifying and mitigating intrusions that pose a risk to the security of information (Albulayhi *et al.*, 2021). Scholars have conducted research on machine learning-based approaches to detect unauthorized individuals in Internet of Things (IoT) networks by analyzing anomalies or employing traffic categorization techniques within intrusion detection systems (Ahmed *et al.*,

2021). Furthermore, the utilization of deep convolutional neural networks has exhibited promise in forecasting and identifying irregularities across diverse fields, emphasizing the significance of these methodologies in real-world scenarios (Meena *et al.*, 2021).

Additionally, Machine Learning (ML) models have been utilized in diverse contexts, such as traffic forecasting, demonstrating the adaptability of these models in effectively processing time-series data for predictive objectives (Zellner *et al.*, 2021). The implementation of intrusion detection systems is crucial in the domain of Internet of Things (IoT) networks to guarantee the integrity and security of the system. This entails emphasizing many security techniques, including cryptography and networking protocols (Abosata *et al.*, 2021). In addition, the application of swarm intelligence algorithms has been proposed for the purpose of intrusion detection in the domain of network security, thereby demonstrating the diverse range of methodologies currently being investigated in this particular area of study (Zellner *et al.*, 2021).

In summary, the current state of research pertaining to network traffic prediction for intrusion detection is characterized by its diverse and ever-changing nature. It places significant importance on harnessing cutting-edge technologies, including machine learning, deep learning, and neural networks, to bolster the security of communication networks. Researchers want to create proactive and effective systems that can mitigate developing cyber threats in today's linked world by combining these creative approaches with traditional intrusion detection methods (Sun *et al.*, 2021).

2. Literature Review

Sarhan *et al.* (2022) have demonstrated that the proposed study exhibits a strong predictive capability. The authors have conducted an investigation into the prediction of traffic matrices. A proposed design has been put out, which incorporates the LSTM Recurrent Neural Network in conjunction with a linear regression model. The experiment was conducted using data from the Abilene network, and the results demonstrated an enhancement in the system's performance.

Elsherif (2018) have introduced a novel framework called LSTM-TPDTNS, which utilizes LSTM to forecast dynamic transport network slicing. The work was conducted in two distinct phases, namely the forecast phase and the bandwidth configuration phase. The system's service quality has been effectively enhanced.

Sethi *et al.* (2020) have introduced a convolutional LSTM model, referred to as ConvLSTM. This model is a fusion of the CNN and LSTM models. In addition, they have employed the Recurrent Neural Network (RNN) to rectify the input data. This study was conducted using the Abilene dataset.

The approach STL-LSTM was proposed by Caminero *et al.* (2019). The seasonal and trend decomposition (STL) has been employed in this study. The researchers have demonstrated that the utilization of this approach enables efficient prediction of base station traffic.

According to Feng et al. (2020). The present study introduces the concept of short-term traffic prediction. Three models based on the LSTM neural network have been proposed and compared. Evidence has demonstrated that dividing the route enhances the precision of forecasting.

Mitsuhashi *et al*. (2021) introduced a network traffic prediction technique that utilizes the firefly swarm optimization algorithm to enhance the convergence speed and learning capability of the BP neural network (GSOBPNN). The objective is to minimize the error in network traffic prediction.

In their study, Rose *et al.* (2021) introduced an enhanced differential evolution backpropagation (BP) algorithm for optimizing the fuzzy neural network (IDEBPFNN) network traffic prediction approach. This was achieved by optimizing the output parameters of the FNN network, resulting in better accuracy in traffic prediction.

In their study, Rajesh *et al.* (2021) introduceda novel approach for network traffic prediction. This method utilizes a dynamic adaptive search step (DASSS) enhanced seeker algorithm (SOA) and an optimized wavelet neural network (WNN). In their study, Ramakrishnan et.al. (2018) employed recurrent neural network (RNN) and its variations, including long short-term memory (LSTM) network and gated recurrent unit (GRU) structure, to examine and forecast network traffic. They demonstrated that RNN is capable of effectively capturing intricate network traffic patterns. The relationship is non-linear and exhibits long-term dependence.

Hwang *et al.* (2020) examined a neural network model called long and short-term memory (LSTM) to forecast network traffic that exhibits non-linear attributes. In their study, Tokuyama et al. (2020) introduced a novel onehot coding technique aimed at enhancing the conventional RNN-VTD model for network traffic prediction.

In their study, Abdulhammed *et al.* (2018)developed a hybrid model that combines a convolutional neural network (CNN) with a recurrent neural network (RNN) for the purpose of predicting Mobile Internet traffic. In a study conducted by Ming et al. (2020), the researchers integrated the Attention mechanism, Residual Neural Network (ResNet), and Recurrent Neural Network (RNN) in order to forecast wireless network traffic.

3. Methodology

The proposed system architecture comprises different components of the system. A detailed description of the proposed system design can be seen in Figure 1.

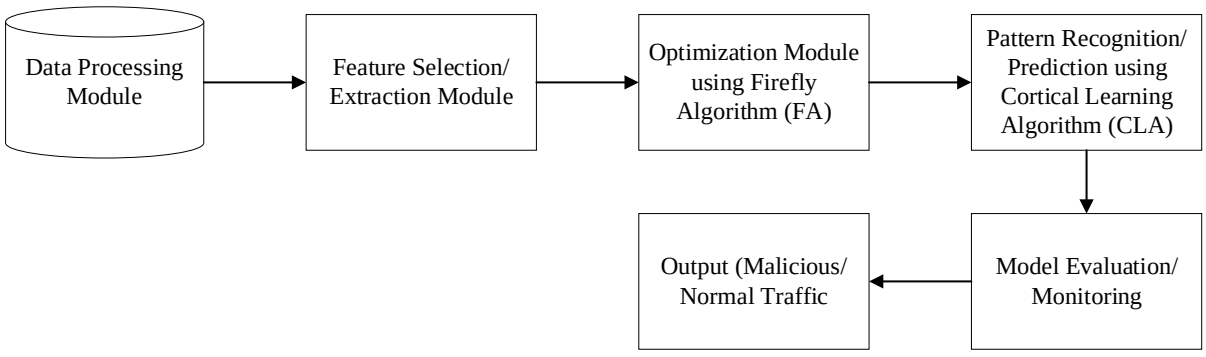

Figure 1: Architecture of the Proposed System

Data Preprocessing Module: This particular component encompasses the processes of data collecting, cleansing, and standardization. Preprocessing is performed on raw network traffic data acquired from diverse sources in order to eliminate noise, address missing values, and standardize the data format for subsequent analysis. Data preparation is a crucial step in ensuring that the input data conform to the requirements of forecasting models.

Feature Selection and Extraction Module: In order to discover the most pertinent features from the preprocessed data that make a substantial contribution to network traffic patterns, feature selection approaches are utilized. In addition, feature extraction techniques can be utilized to convert unprocessed data into more significant representations. This module enhances the effectiveness of forecasting models by reducing dimensionality and prioritizing crucial input variables.

Firefly Algorithm (FA): The Firefly technique (FA) is a metaheuristic optimization technique that utilizes the flashing behavior of fireflies to optimize the parameters of forecasting models. In the domain of network traffic forecasting, the utilization of Fuzzy Arrays (FA) can be employed to optimize the weights and parameters of the forecasting model, including neural network designs or ARIMA parameters. The performance of the forecasting model is enhanced by the ability of FA to explore the search space and converge towards the global optimum.

Cortical Learning Algorithm (CLA): The CLA algorithm, which draws inspiration from neuroscience concepts, is a machine learning technique that enables the acquisition of temporal and spatial patterns from streaming data. CLA can be employed in network traffic forecasting to capture intricate patterns and correlations that are inherent in network traffic data over a period of time. CLA dynamically adjusts its predictive model by imitating the behavior of cortical neurons, allowing it to adjust to varying network conditions and enhance prediction accuracy.

Hybrid Integration Layer: A unified forecasting framework is facilitated by this layer, which enables the integration of FA and CLA components. This approach facilitates the integration of the optimization skills of Fuzzy Adaptive (FA) and the learning capabilities of Conditional Randomization (CLA) in order to collaboratively tune the parameters of the forecasting model, while also leveraging previous network traffic data for learning purposes. This integration utilizes the synergistic advantages of FA and CLA to improve the precision and resilience of predictions.

Evaluation and Performance Monitoring Module: The performance of the forecasting model is assessed by this component through the utilization of measures such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). This facilitates the assessment of the precision and dependability of the forecasts, allowing for ongoing surveillance and enhancement of the forecasting system.

Algorithm of the hybrid Model

1**. Initialization:**
Initialize FA parameters:
- Population size (N)

- Maximum iterations (max_iter)
- Convergence threshold (epsilon)

Initialize CLA parameters:
- Neural network architecture (e.g., number of layers, neurons per layer)
- Learning rate, convergence criteria, etc.

Load and preprocess network traffic data.

2. **Training Phase:**

a. Firefly Algorithm Optimization:

```
Initialize random population of fireflies
while not converged or not reached maximum iterations:
    for each firefly in the population:
        Calculate attractiveness with other fireflies:
            attractiveness(i,j) = beta * exp(-gamma * distance(i,j)^2)
        Update firefly position based on attractiveness and distance:
            newPosition = current_position + step_size * (rand() - 0.5)
    Evaluate fitness of each firefly based on objective function (prediction error)
    Select the best fireflies based on fitness
    Update parameters of the neural network model using selected fireflies
end while
```

b. Cortical Learning Algorithm Training:

```
Initialize neural network weights and biases
while not converged or not reached convergence criteria:
    for each training sample:
        Feedforward input through the network:
            output = activation_function(weights * input + biases)
        Compute prediction error:
            error = actual_output - predicted_output
        Update synaptic connections using CLA learning rules:
            weight_update = learning_rate * error * input
            new_weights = old_weights + weight_update
    end for
end while
```

3. **Prediction Phase:**

Input latest network traffic data
Obtain forecasted values from both optimized neural network model and CLA model
Optionally ensemble predictions using weighted averaging or other fusion technique

4. **Evaluation and Performance Monitoring:**

Evaluate forecasting performance using metrics (MAE, RMSE, MAPE)
Monitor model's performance and adjust parameters as needed

5. **Repeat Training and Prediction:**

Periodically retrain models with updated data
Repeat prediction phase for new data.

6. **Final Model Deployment:**
Deploy trained hybrid model for real-time network traffic forecasting.

4. Results

4. Experimental Results

We conducted an experiment on Jupyter Notebook, the experimental results is made up of two phases. The phases of the experiments are the Exploratory Data Analysis (EDA) and the training of the Random Forest Classifier for the detection of malicious traffics on network systems.

4.1 Exploratory Data Analysis (EDA)

Exploratory data analysis (EDA) on the dataset in other to have a better insight into the NSL-KDD dataset. In Figure 2, a correlation was carried out on the dataset features. This was carried out using seaborn library in python. Next, an EDA was conducted to know if the dataset is imbalanced. According to the findings presented in Figure 3, it is evident that the dataset exhibits an imbalance, as the number of instances in each class is not equal. In order to address this issue, a random oversampling technique was employed in the Python programming language. This technique involved increasing the number of instances in the minority classes to match that of the majority class. The counplot of the balanced data can be seen in Figure 4 The last part of the EDA was to find the ten most important features of the dataset. The achieved using Isolation Forest. This was done by performing ranking on the features of the dataset. From the ranking, the ten most important features can be seen in Table 1 and Figure 5.

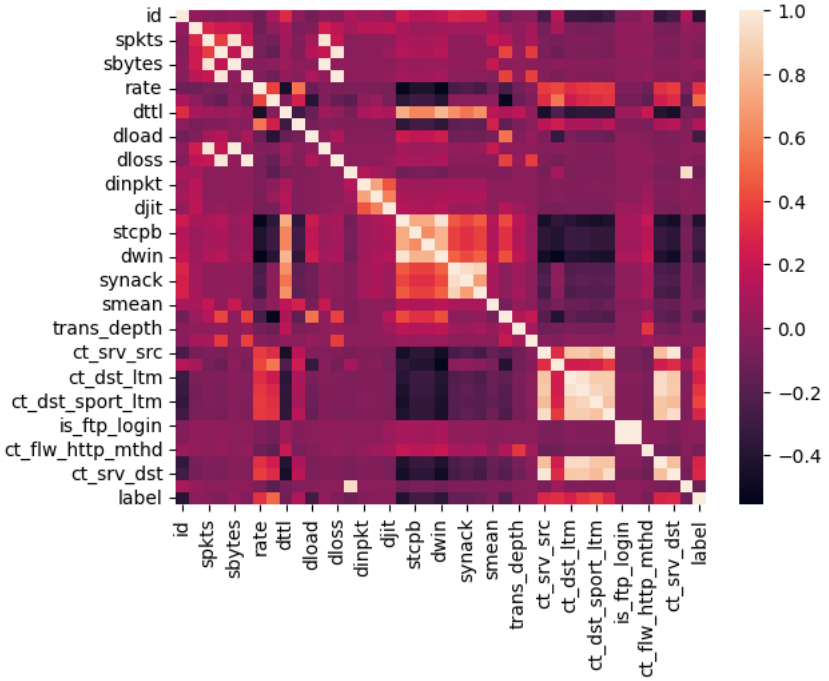

Figure 2: Correlated Features

The correlation matrix shows if there exist a relationship between features of the dataset.

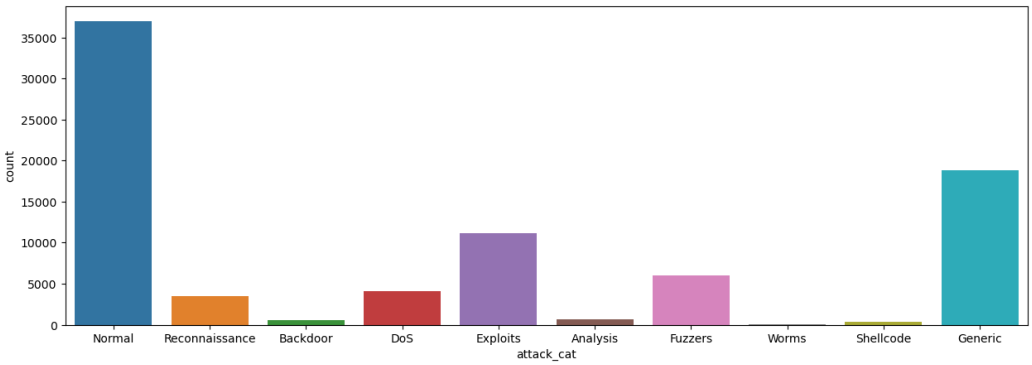

Figure 3: Countplot of the Imbalanced Data

This shows that the dataset is imbalanced. This is because the classes of the dataset are not equal.

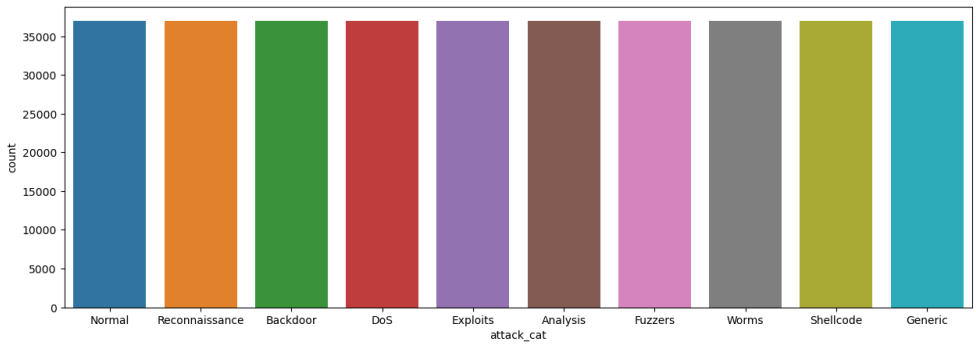

Figure 4: Countplot of the Balanced Data

This shows that the classes of the dataset have equal number of instances.

Table 1: Most Ten Important Features

	Feature	Important_Features
0	id	0.235239
1	sbytes	0.098822
2	sload	0.053006
3	ct_dst_sport_ltm	0.050069
4	smean	0.045883
5	service	0.044786
6	ct_srv_dst	0.034816
7	ct_dst_src_ltm	0.030366
8	dpkts	0.026507
9	ct_src_dpoutItm	0.026507

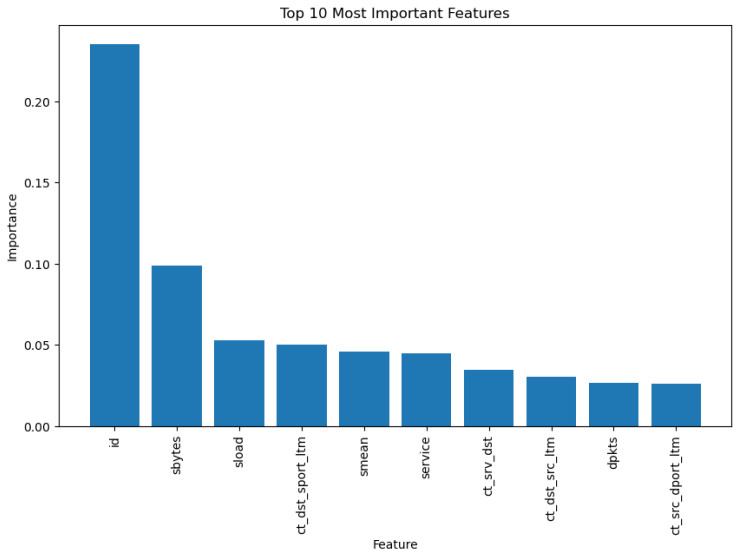

Figure 5: Top ten important features

This shows the ranking of the dataset features.

4.2 Model Training with Cortical Learning

In the model training process utilizing the hybrid approach of firefly algorithm and cortical learning for network traffic forecast, the first step involves initializing the parameters for both algorithms. The firefly algorithm parameters include the population size, attractiveness coefficient, and maximum iteration count. These parameters govern the exploration and exploitation capabilities of the fireflies in the optimization process. Simultaneously, for cortical learning, parameters such as the synaptic plasticity rates, learning rates, and memory decay factors are set to regulate the adaptation and memory retention of the cortical neurons. The integration of the two algorithms requires careful tuning of the hybridization parameters, determining the influence and interaction strengths between the firefly and cortical learning components. The training process involves iterative optimization, where the firefly algorithm refines the initial parameter space, and the cortical learning algorithm adjusts synaptic weights based on observed patterns in network traffic data. The model was continuous fine-tuning of these parameters is essential to achieve an optimal balance between exploration and exploitation, enabling the hybrid model to effectively capture and forecast complex patterns in network traffic dynamics. The outcomes of the model for both the training and testing can be seen in Table 2, Figure 6 and 7. The model was evaluated and its performance was validated using a separate test dataset. The evaluation metrics employed in this study are the categorization report and the confusion matrix. The classification report is presented in Figure 8 while the confusion matrix is displayed in Figure 9.

Table 2: Training Process of the LSTM for faults detection in autonomous vehicles

Epoch 1/10
1204/1204 [==============================] - 10s 5ms/step - loss: 0.3612 - accuracy: 0.8876 - val_loss: 0.1745 - val_accuracy: 0.9457
Epoch 2/10
1204/1204 [==============================] - 5s 5ms/step - loss: 0.1432 - accuracy: 0.9552 - val_loss: 0.1253 - val_accuracy: 0.9596
Epoch 3/10
1204/1204 [==============================] - 5s 5ms/step - loss: 0.1096 - accuracy: 0.9641 - val_loss: 0.1021 - val_accuracy: 0.9655
Epoch 4/10
1204/1204 [==============================] - 5s 4ms/step - loss: 0.0874 - accuracy: 0.9723 - val_loss: 0.0855 - val_accuracy: 0.9715
Epoch 5/10
1204/1204 [==============================] - 5s 4ms/step - loss: 0.0727 - accuracy: 0.9772 - val_loss: 0.0746 - val_accuracy: 0.9754
Epoch 6/10
1204/1204 [==============================] - 5s 5ms/step - loss: 0.0625 - accuracy: 0.9814 - val_loss: 0.0708 - val_accuracy: 0.9744
Epoch 7/10
1204/1204 [==============================] - 5s 4ms/step - loss: 0.0558 - accuracy: 0.9826 - val_loss: 0.0604 - val_accuracy: 0.9829
Epoch 8/10
1204/1204 [==============================] - 5s 5ms/step - loss: 0.0516 - accuracy: 0.9849 - val_loss: 0.0552 - val_accuracy: 0.9826
Epoch 9/10
1204/1204 [==============================] - 5s 5ms/step - loss: 0.0491 - accuracy: 0.9854 - val_loss: 0.0619 - val_accuracy: 0.9825
Epoch 10/10
1204/1204 [==============================] - 5s 4ms/step - loss: 0.0478 - accuracy: 0.9961 - val_loss: 0.0540 - val_accuracy: 0.9940
CPU times: total: 1min 55s
Wall time: 58.8 s

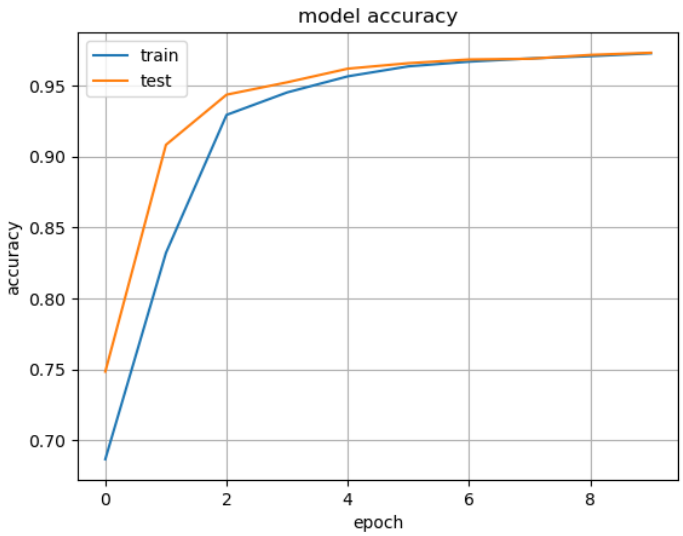

Figure 6: Graphical Representation of the model for training and testing

This shows the result of the model for training and testing.

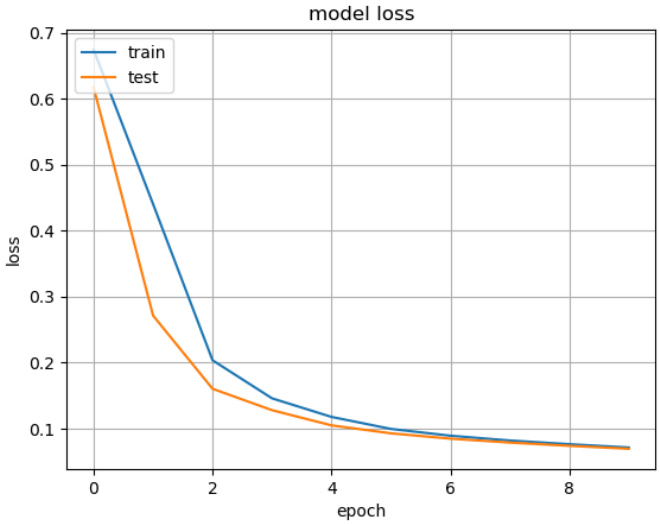

Figure 7: Graphical Representation of the model loss for training and testing

This shows the loss values obtained by the model during training and testing.

	precision	recall	f1-score	support
Normal	1.00	1.00	1.00	677
Backdoor	1.00	1.00	1.00	583
Analysis	1.00	1.00	1.00	4089
Fuzzers	1.00	1.00	1.00	11132
Shellcode	1.00	1.00	1.00	6062
Reconnaissance	1.00	1.00	1.00	18871
Exploits	1.00	1.00	1.00	37000
DoS	1.00	1.00	1.00	3496
Worms	1.00	1.00	1.00	378
Generic	1.00	1.00	1.00	44
accuracy			1.00	82332
macro avg	1.00	1.00	1.00	82332
weighted avg	1.00	1.00	1.00	82332

Figure 8: Classification Report of the cortical learning model

The result of the classification matrix shows the evaluation of the model in detecting the network traffic.

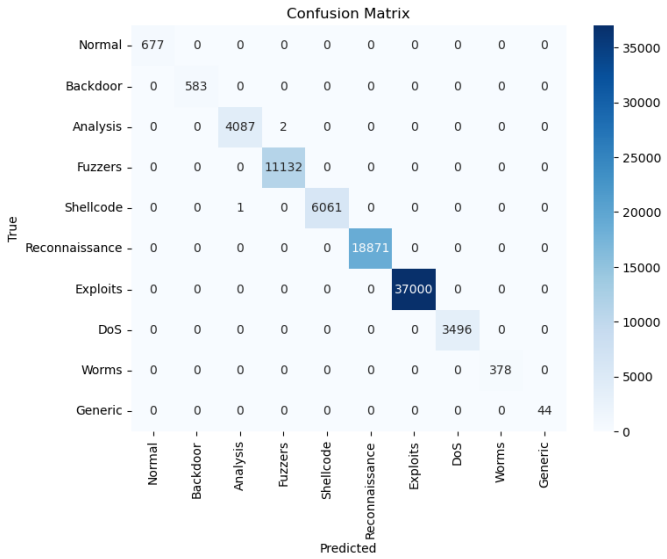

Figure 9: Confusion Matrix of the cortical learning model

This shows the number of true and false prediction of the cortical learning model on network system.

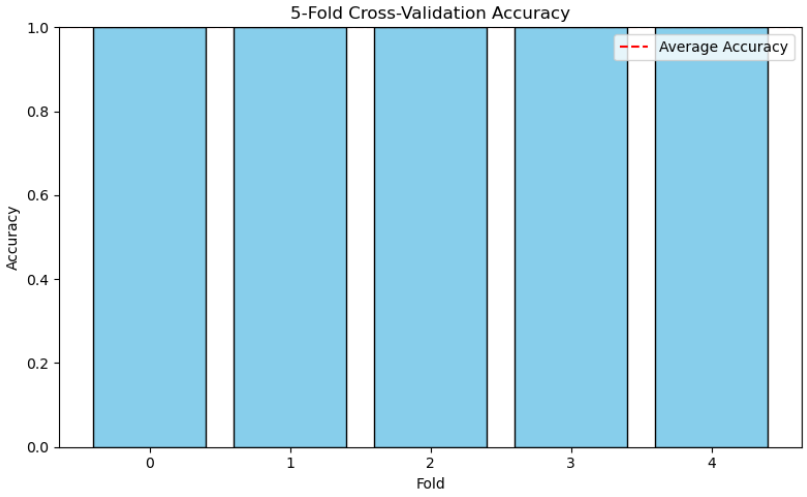

Figure 10: Five-Fold Training Validation

This shows the evaluation of the model using five-fold cross validation.

4.3 Comparison with other Existing System

This sub section describes the evaluation of the proposed system with other existing systems. The evaluation was carried out in terms of the accuracy of the models. This can be seen in Table 3 and Figure 4.11.

Table 3: Comparison With Other Existing System

Authors	Title	Accuracy (%)
Indrasiri *et al.* (2022)	"Malicious traffic detection in iot and local networks using stacked ensemble classifier"	98.5
Alshammari and Aldribi (2021)	"Apply machine learning techniques to detect malicious network traffic in cloud computing"	94
The proposed method		99.9

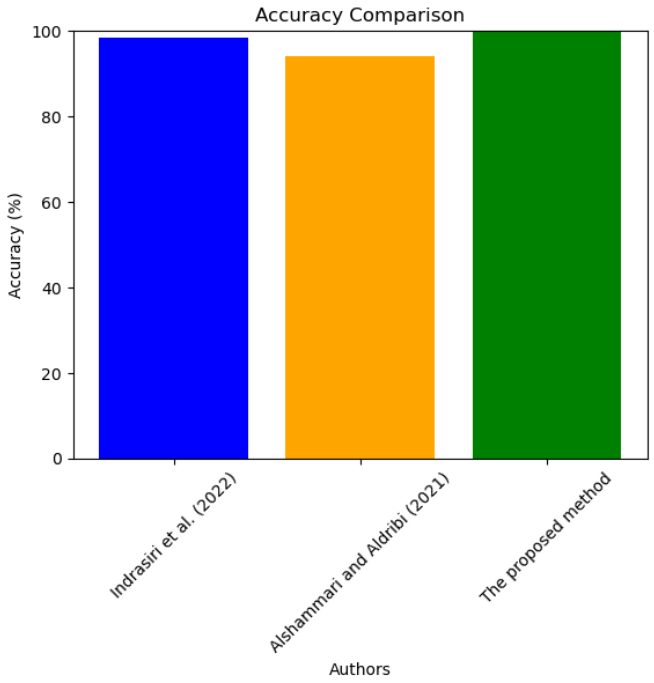

Figure 11 Evaluation with other existing systems

5. Discussion of Results

From the experiment conducted the correlation matrix of associated features is depicted in Figure 2, as observed in the conducted experiment. The correlation matrix is employed to assess the presence of a relationship between attributes within a given dataset. The correlation matrix reveals that some features within the dataset exhibit correlation. Figure 3 displays the countplot representing the imbalanced data. According to the data presented in Figure 3 it is evident that the normal class exhibits the greatest bar, while the generic class follows closely behind. Failure to address this issue may result in the occurrence of overfitting during the model training process. Figure 4 depicts the countplot of the balanced dataset, whereby each class is represented by an equal number of observations. Table 1 presents the initial 10 columns of significance within the dataset, as determined by the random forest model's ranking for the purpose of selecting the most crucial attributes. The table illustrates the characteristics that exert a greater impact on the dataset. Figure 5 presents a visual depiction of the key features. From Figure 5 that the id feature holds the highest influence on the dataset, followed by the bytes feature.
Figure 6 shows the accuracy of the model against epoch for both training and validation data. The graphical representation shows that the accuracy of the model during training and validation is 99%. Figure 4.6 shows the loss values of the model during training and validation data. The graphical results shows that the model had a loss value below 0.1 for both training and validation data.

Figure 7 represent the classification report of the suggested model in relation to the identification of malicious traffic within a network system. The precision score, F1-score, recall

score, and accuracy of each class in the data may be observed from the classification report. These scores indicate a high level of performance, with a value of 99.99% that can be estimated as 100%.

Figure 8 depicts the confusion matrix, which serves the purpose of illustrating the frequency of accurate predictions made by the model in identifying malicious network traffic within the system. The confusion matrix reveals that the suggested model exhibits minimum occurrences of false positive and false negative values. What is the prevalence of false positive and false negative rates at 0.001%? Figure 9 illustrates the mean accuracy of the model obtained during a five-fold cross-validation procedure. The five-fold cross-validation method was employed to evaluate the model's performance across five distinct iterations. It is evident that the model consistently attained a 99.99% accuracy rate at each training stage. The average accuracy is 99.99%.

Figure 10 shows the comparison of the system with other existing systems. The result shows the that the proposed hybrid model outperforms the existing systems.

6. Conclusion

A cortical learning algorithm has been used to forecast network traffic, enhancing efficiency and accuracy. The first objective was to acquire network traffic data from UCL, which was then cleaned and pre-processed using Pandas library. The second objective involved extracting important features using isolation forest, which identified anomalies and prevented them. The third objective was to develop a model using the extracted data, demonstrating the effectiveness of cortical learning algorithms in capturing complex patterns and dependencies, making it a valuable tool for real-time traffic flow predictions.

References

Abdulhammed, R., Faezipour, M., Abuzneid, A., & AbuMallouh, A. (2018). Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic. *IEEE sensors letters*, *3*(1), 1-4.

Abosata, N., Al-Rubaye, S., Tsourdos, A., & Emmanouilidis, C. (2021). Internet of things for system integrity: a comprehensive survey on security, attacks and countermeasures for industrial applications. Sensors, 21(11), 3654. https://doi.org/10.3390/s21113654

Ahmed, K., Tahir, M., Habaebi, M., Lau, S., & Ahad, A. (2021). Machine learning for authentication and authorization in IoT: taxonomy, challenges and future research direction. Sensors, 21(15), 5122. https://doi.org/10.3390/s21155122

Albulayhi, K., Smadi, A., Sheldon, F., & Abercrombie, R. (2021). IoT intrusion detection taxonomy, reference architecture, and analyses. Sensors, 21(19), 6432. https://doi.org/10.3390/s21196432

Caminero, G., Lopez-Martin, M., & Carro, B. (2019). Adversarial environment reinforcement learning algorithm for intrusion detection. *Computer Networks*, *159*, 96-109.

Elsherif, A. (2018). Automatic intrusion detection system using deep recurrent neural network paradigm. *Journal of Information Security and Cybercrimes Research*, *1*(1), 21-31.

Feng, J., Shen, L., Chen, Z., Wang, Y., & Li, H. (2020). A two-layer deep learning method for android malware detection using network traffic. *IEEE Access*, *8*, 125786-125796.

Hwang, R. H., Peng, M. C., Huang, C. W., Lin, P. C., & Nguyen, V. L. (2020). An unsupervised deep learning model for early network traffic anomaly detection. *IEEE Access*, *8*, 30387-30399.

Meena, S., Dhaka, V., Sinwar, D., Kavita, ., Ijaz, M., & Woźniak, M. (2021). A survey of deep convolutional neural networks applied for prediction of plant leaf diseases. Sensors, 21(14), 4749. https://doi.org/10.3390/s21144749

Mitsuhashi, R., Satoh, A., Jin, Y., Iida, K., Shinagawa, T., & Takai, Y. (2021). Identifying malicious dns tunnel tools from doh traffic using hierarchical machine learning classification. In *Information Security: 24th International Conference, ISC 2021, Virtual Event, November 10–12, 2021, Proceedings 24* (pp. 238-256). Springer International Publishing.

Rajesh, L., & Satyanarayana, P. (2021). Evaluation of machine learning algorithms for detection of malicious traffic in scada network. *Journal of Electrical Engineering & Technology*, 1-16.

Rose, J. R., Swann, M., Bendiab, G., Shiaeles, S., & Kolokotronis, N. (2021, June). Intrusion detection using network traffic profiling and machine learning for IoT. In *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)* (pp. 409-415). IEEE.

Sarhan, M., Layeghy, S., & Portmann, M. (2022). Towards a standard feature set for network intrusion detection system datasets. *Mobile networks and applications*, 1-14.

Sethi, K., Sai Rupesh, E., Kumar, R., Bera, P., & Venu Madhav, Y. (2020). A context-aware robust intrusion detection system: a reinforcement learning-based approach. *International Journal of Information Security*, *19*, 657-678.

Sun, W., Tang, M., Zhang, L., Huo, Z., & Shu, L. (2020). A survey of using swarm intelligence algorithms in IoT. Sensors, 20(5), 1420. https://doi.org/10.3390/s20051420

Zellner, M., Abbas, A., Budescu, D., & Galstyan, A. (2021). A survey of human judgement and quantitative forecasting methods. Royal Society Open Science, 8(2). https://doi.org/10.1098/rsos.201187